SCons and Windows - a Log

Includes status, proposed strategies. Newest entries are at the top. See also a (Windows-specific) to-do list.

July 8,2009

Windows users with Studio 2003 or later can get experience with SCons and GoGui just by installing them (SCons 1.2.0 or later; GoGui 0.9.5 or later). Check out ScienceTools directly from CVS (and tell GoGui where you put it) or check it out from within GoGui. With this set-up SCons will not make project files.

Windows alpha testers for project file support will need to obtain

1. Studio 2008
2. GoGui 0.9.5 or later (and a couple required libraries if you don't already have them)
3. SCons 1.2.0.d20090223

Then see instructions on page 8 (titled "Windows only") in this attachment.

~~For each package the SConscript file must be modified slightly to invoke the tool registerTargets rather than registerObjects.~~ **This has now been done for all packages in ScienceTools** New, converted SConscript files have been committed and the packages are all tagged. If you start with source from a recent ST LATEST build you don't have to worry about this. Details of the differences are described below for the curious. Others should skip to the section **GoGui Hints** below.

Technical details

registerTargets has a different interface. First of all, registerTargets makes finer distinctions between target types. swig libraries are a category of their own, as are ROOT-using static libraries and ROOT-using shared libraries. For each category of library or executable target (e.g. static library, swig library, test app, etc.) instead of passing just a list of targets you pass in a list of pairs [*target*, *environment*] where *environment* is the environment used to create the target somewhere earlier in the SConscript file. Here is an example of the changes needed for the facilities package:

```
#Old form of register, using registerObjects
#progEnv.Tool('registerObjects', package = 'facilities',
#             libraries = [facilitiesLib, lib_pyFacilities],
#             testApps = [test_time, test_env, test_Util],
#             includes = listFiles(['facilities/*.h']))

#New, using registerTargets
progEnv.Tool('registerTargets', package = 'facilities',
             libraryCxts = [[facilitiesLib, libEnv]],
             swigLibraryCxts = [[lib_pyFacilities, swigEnv]],
             testAppCxts = [[test_time, progEnv], [test_env,progEnv],
                            [test_Util,progEnv]],
             includes = listFiles(['facilities/*.h']))
```

GoGui hints

Alpha users should go to the **utility options** item in the options menu to make sure the correct SCons will be used. Go also to **build options** and select vc9 for the compiler.

You can make use of the "Specify build target" input at the bottom of the GoGui window to build just what you need for project files. The input accepted there depends on which package is selected in the Navigator panel. For our purposes, select the top container. Then make the following targets by typing the name in the input area, followed by **Enter** (or click on the target icon):

- **setup** This will create the file _setup.bat, used to set environment variables when Studio is invoked. (see screenshot)
- **install** Installs includes, xml files, python files, etc.
- **studio** Creates all project and solution files.

See instructions below (entry for June 18) on how to invoke Studio for a particular package.

June 18, 2009

The problem is not VSlauncher, at least not exclusively VSlauncher. Hard-code the path to Studio 2008 devenv as the command to be issued has the same problem: devenv shows up as a running process in the Task Manager, but doesn't put up any windows until GoGui exits. However a command like the following works:

start "any title" **path-to-VSlauncher path-to-sln-file**

The title string is necessary even though it's not actually used for anything and is supposedly optional. **path-to-VSlauncher** has to be quoted since it typically includes spaces, so if there is no title start will think that **path-to-VSlauncher** is the title rather than the command.

If a package is selected, you can invoke Studio with the solution file for that package from the toolbar. You'll be prompted to select the proper variant (e.g. debug versus opt or different compilers) if the solution file exists for more than one. Alternatively, select the top-level container and use the browser to navi gate to the desired solution file (under studio folder) and right-click. In either case a new Studio instance should appear soon.

June 17, 2009

Invocation of devenv is, frustratingly, not quite right. I would like to use the Windows VSlauncher program which, when given a solution file, finds the right devenv and invokes it. That doesn't quite work. VSlauncher gets started all right, but doesn't invoke devenv until I quit GoGui! However, if I just call devenv (Studio 2003 devenv, since that's what is in my path) itself from GoGui, using the very same underlying process creation and what-not, it does the right thing.

June 16, 2009

Creation of command window from GoGui with proper set-up is working. Might not always handle odd cases involving multiple variants properly.

June 15, 2009

May have acceptable implementation for creating Windows set-up file from GoGui. Now can revisit creation of dos window and invocation of devenv from GoGui.

June 5, 2009

About 20 ScienceTools packages have modified SConscript files (in my local sandbox only, not committed) which are compatible with making project files (for VS 2008 only).

Original static library implementation wasn't quite right. That has been fixed. However, there is a lot of rebuilding going on in Studio which shouldn't be necessary. Any project depending on a static library wants to recompile all source, whether it belongs to the (already-build) static library or only to the project.

Other problems with SCons on Windows include:

- too many files included in the project file sources section. Not clear whether or not this is causing any problems with the build or not (e.g., the unnecessary recompiling noted above). When I remove them by hand from the project file, behavior is the same.
- one known example of a ScienceTools package (evtbin) which compiles nicely under Studio 2003 but fails with Studio 2008. There may be more, another impediment to moving to Studio 2008.
- ~~weirdness when building st_graph. After static library and test program (test_st_graph.exe) are built, SCons builds test_st_graph.lib. Who ordered that?~~ [June 9 update: turns out this has been happening for a long time with CMT builds as well as SCons. See e.g. RM output for v9r14 (static library) or v9r15 (dll).]

June 2, 2009

After a lot of experimentation with building shared libraries rather than static for certain ST packages to avoid having to support static root libraries, I gave it up. There were too many mysteries. Instead I've added root static libraries to the list of supported project types.

May 18, 2009

I forgot about static libraries, yet another target type needing a somewhat different project file. The project file types currently implemented are

1. static library
2. 'plain' shared library
3. wrapper shared library (can be loaded from python)
4. root shared library (can be loaded from interactive root)
5. executable

Gaudi component libraries may require yet another slight variation.

Heather made CLHEP for vc90. All externals needed for ScienceTools on vc90 are now built!

May 15, 2009

Submitted a bug report to SCons. It seems that on Windows, when using variant builds, which we do, and SCons in interactive mode, which GoGui does, SCons doesn't notice when a source file is changed. Could be it's doing its checks on the copy in the variant directory rather than the original source. A similar bug was already reported months ago and the issue is marked 'FIXED' but it looks like they didn't fix it for all platforms.

May 11, 2009

Finished off rootcint support at the end of last week. This completes the major known pieces of the Windows/SCons puzzle but there are still some small holes and the code (both Windows-specific and our generic local SCons support) is in desperate need of clean-up. See this new, likely incomplete Scons tools to-do list.

May 5, 2009

Current activities are:

- Support for rootcint in project files (in progress)
- Interactions with a real, live user: Tracy! Follow links to see the first fruits of our collaboration: draft GoGui desktop icon (ico or png format).

April 21, 2009

Can write usable project file for swig library. See an example for the facilities swig library.

In order to use Studio with the project files SCons writes, one must have a suitable environment; for example, PATH must include external library directories. Still more set-up is needed to use swig from Studio. This has been handled by

- enhancing to external.scons to make more information concerning the externals accessible to other tools
- adding code to generateScript.py to write a file _setup.bat (see what gets written for a toy installation with just two packages and minimal set of external libraries) which can be run before invoking Studio.

April 10, 2009

Dependencies are in solution files as they should be (at least, they're correct for facilities and xmlBase). See e.g. solution file for xmlBase and screen shot from Studio while using it.

April 2, 2009

- Project and solution files are now all installed in directory studio_variantName_ under root, analogous to bin, lib, etc.
- Split registerObjects tool into two new tools.
    - registerStuff (working name only) is very similar to original registerObjects except interface is changed a bit. Libraries, binaries and testApps all are specified along with appropriate construction environment. package SConscript files need only call registerStuff. It will then call makeStudio if platform is Windows.
    - makeStudio makes project and solution files; that is, it invokes MSVSProject and MSVSSolution. It supplies set of required libraries for each project file in the solution.

Still to do: mods to msvs.py

March 27, 2009

Fixed up externals.scons (private copy) so that it builds swig wrapper libraries without error on both Linux and Windows (this was already working on SLAC Linux, but failed on my laptop when I copied the SLAC installation of SWIG there, probably due to hard-coded paths created when SWIG was built which were useless without access to SLAC afs).

Starting to figure out what will be needed to include all necessary projects in a solution file (not just the ones in the package) and keep track of their dependencies. See discussion in to-do list.

March 23, 2009

Current stuff makes a per-package solution file with absolute paths for referenced project files, hence works no matter where the solution file is. Explicitly install it in directory *pkg\visual-variant* (e.g., facilities\Visual-vc90-Debug\facilities.sln).

Added code to define INST_DIR in _setup.bat. Can now run (as well as build) programs like test_Ifile in package xmlBase which depend on this environment variable. The Studio debugger is able to find and display source from facilities as well as xmlBase. But Studio constantly thinks it needs to rebuild things it doesn't actually need to rebuild.

This screen shot shows new arrangement of "stop" buttons (lower right) and build-specific-target widget.

March 16, 2009

Added some code to tool generateScript.py to write an additional file, _setup.bat, on Windows only. It sets up PATH variable so that Studio can usefully be run with the SCons-generated project files. _setup.bat needs more work; it doesn't yet establish other environment variables which _setup.vbs sets. (The problem with _setup.vbs is that it creates the new environment in a separate process, then exits before anything useful can be done with it.) _setup.bat must be created by the developer (not just imported as part of an install) because it contains absolute file paths.

March 12, 2009

Xerces provides a Visual Studio 8.0 binary build of version 2.8.0. We have been using 2.7.0, but 2.8.0 is supposed to be backwards compatible. I installed these binaries and they appear to work with Visual Studio 9.0. This means my development installation can be somewhat more realistic, with two packages (facilities and xmlBase), one of which truly depends on an external library (references its include files; needs to link to its library).

March 10, 2009

Addendum to March 9th entry: have confirmed that, when the PATH variable is augmented to include SCons lib directory, applications like test_Util.exe will run inside Studio.

March 9, 2009

Can now generate a project file for facilities library which almost does the right thing (everything except for putting outputs where I thought I asked it to) and project files for facilities apps which link but don't run, probably because PATH doesn't include directory of facilities.lib

Concerns in previous entry have not gone away. I have made some guesses about where build products should go and to what extent the two build systems share files, but

- Haven't quite implemented what I'm after
- Am not confident these guesses can be extrapolated to cover a realistic development installation with many packages and possibly an override directory.

March 5, 2009

Accomplished some clarification by adding more information to variant names. SCons variant now incorporates compiler (e.g. instead of redhat4-i686-32bit-Debug on Linux will now get redhat4-i686-32bit-gcc34-Debug; on Windows now get XP-i386-32bit-vc90-Debug). Project files also include a name called the 'variant'. It now looks just like the SCons variant except it starts with **Visual**, e.g. Visual-vc90-Debug.

However problems involving directory structure, interactions between SCons builds and Studio builds are looming. The work areas - where intermediates like object files get built - are distinct and probably should remain so. But what about the install area? SCons builds both refer to it when searching for include files or libraries and write to it. If link commands issued by Studio are to be roughly analogous to those used by SCons, it has to do the same thing. In this case, should it use the same install area or a separate one? (I would think the same.) How do I tell Studio to copy libraries to the install area? What about public sources; that is, the files that go in the top-level include, xml, etc., directories?

March 4, 2009

Inching forwards.. Project file produced now can be used by Studio to make a .def file (needed at link time to export everything we want exported) which looks more or less reasonable, though smaller than the one SCons makes with its "native" build. However the prelink command is probably still not quite right. It's using the objects build by SCons as input rather than the objects Studio built. ~~Should be able to fix~~ Have fixed this with more tinkering, but the total amount of tinkering going on is unsettling at best.

March 2, 2009

Found a useful variable kept by SCons for compiler options and made use of it in writing out project file so that all compile options, most importantly including INCLUDE directories, are handled in a general fashion. Resulting project file for facilities can be used to compile all .cxx source. There are still several obstacles to using it to build the library.

Made a new module, users/jrb/minimalDev, in the CVS respository for ongoing work on SCons machinery (SConstruct, registerObjects.py, etc.) for Windows. Since it's dependent on SCons-1.2.0.d20090223 it's not compatible with production versions of these files.

February 26, 2009

Installed SCons release 1.2.0.d20090223 - much easier to deal with than working off a development branch.

Committed some minor changes to SConstruct for support of Windows:

- added line to create manifest files
- added to and rationalized compiler option lines
- added --vc9 option.

February 24, 2009

Have installed VS 2008 Professional. SCons finds it correctly when --vc9 option is used. Problems detected so far in the project and solution files that get written [and estimate of difficulty to fix] include

1. Extra stuff at the end of the .vcproj file and the .sln file which appears to be for the private use of SCons, as SCons central expects these files to be used: that is, coerce Studio into calling SCons to do the actual build. It may be harmless (or maybe not; it might have accounted for some of the strangeness I saw yesterday) but, for the kind of project files we're aiming for, it's certainly not useful. Eliminate it. [easy]
2. Solution file version should be 10.0, not 9.0 [easy]
3. Warnings when I try to build the project about incompatible compile options [easy to medium]
4. No user include paths [medium or worse]

There are certainly many more; I just haven't gotten there yet.

One piece of good news: the code in the vs_revamp branch has been merged into the trunk and is available in release 1.2.0.d20090223.

February 23, 2009

First attempts to bring up VS 2008 Express on project and solution files generated by SCons (with Tracy's additions) have been somewhat bizarre: VS claims it needs to convert them. It makes a new solution file and leaves the project file alone. However, if I exit and restart, it doesn't like the solution file it just made and claims the project file has syntax problems! I'm in the process of installing VS 2008 Professional, in hopes that it will behave better, and also because Professional is the real target environment, not Express.

February 12, 2009

Can now build and run with 9.0. Link problem for debug was solved by specifying compile flag /MDd rather than /MD (thanks, Heather, for the hint). The inability to run was because manifest files created by the 9.0 linker were not getting installed. When I copied them to the proper place, all was well. Navid found mention of a variable WINDOWS_INSERT_MANIFEST in the SCons documentation which, when set to 'true', causes the manifest files to be installed automatically.

February 11, 2009

Was able to successfully build facilities non-debug, but couldn't run applications. Error message was "This application has failed to start because MSVCR90.dll was not found".

MSVCR90 (and a couple other libraries with similar names) are part of the Express 9.0 distribution. On my machine they can be found in directory

```
C:\Program Files\Microsoft Visual Studio 9.0\VC\redist\x86\Microsoft.VC90.CRT
```

The debug versions are in a nearby Debug_NonRedist directory. I copied them all to C:\WINNT\system32, since that's where similar libraries for other Studio versions are, and rebuilt the applications, but that just changed the error to R6034: "An application has made an attempt to load the C runtime library incorrectly. Please contact the applicatins's support team for more information." ..not too helpful since I *am* the application's support team. And I still can't even link debug.

February 10, 2009

Windows server problems of yesterday were addressed. Using vs_revamp code from GoGui, can now

- build facilities package using Studio 7.1
- *almost* build facilities using Studio 9.0

Before I started on the vs_revamp gambit, I could not compile facilities with Studio 9.0 because the include path was missing a directory containing some system include files. That problem is fixed; all the sources in facilities compile successfully. The build now fails while trying to link the library:

```
 Creating library facilities-02-18-02\build\XP-i386-32bit-Debug\facilities.lib
and object facilities-02-18-02\build\XP-i386-32bit-Debug\facilities.exp
Scheduler.obj :
error LNK2019: unresolved external symbol __imp___CrtDbgReportW referenced..
```

February 9, 2009

Checked out the vs_revamp branch from the SCons code repository. Original plan was to use my existing 1.2.0 SCons installation and copy the new Windows stuff from vs_revamp into our site_tools/site_scons directory, but that failed in a mysterious way. Running vs_revamp in place according to instructions in a README file worked, but was not immediately compatible with invoking it from GoGui. Next I tried installing vs_revamp just as one would install the trunk (Subversion-speak for main branch) but couldn't get that to work either. By writing a little .bat file wrapper I am now able to run the vs_revamp version of SCons from GoGui; that is, I can get as far with it as I can by invoking it from the command line. That isn't very far at the moment because I can't reach the SLAC V disk, where the external libraries are. (There was a notice about problems with Windows servers today which, in theory, should not have touched any of the resources I need.)

Summary: After quite a bit of thrashing, net result is a small step forward.

February 2, 2009

The SCons developers are working on a better way to determine proper environment on Windows, depending on VS version. The idea is to find the appropriate set-up file (called something like vsvars32.bat), run it in a special subprocess starting with minimal environment and recover the output environment. This sounds like a much more robust and less-convoluted approach which we should certainly follow. I'm looking into how to get the relevant files, which exist only in a developement branch in their code repository.

January 30, 2009

With a newer version of VS installed on my machine, 7.1 (= VS 2003) is no longer the default. Had to make a couple small changes to SConstruct and to GoGui to recover the ability to build with 7.1 when that is not the default VS version.

Integrating Tracy's work into a local (in site_scons/site_tools) version of the msvs builder is easy; adding support for VS 9.0 turns out to be more difficult than expected. The SCons version of msvs and another related builder, msvc, includes a lot of code specific to various VS versions to find where Studio is installed, what belongs in include and lib path, etc. These things are different for each Studio version. Although VS 8.0 and 9.0 are similar in some ways, including project file format, the directory layout and registry layout are different. New code is needed to handle the 9.0 layout.

Heather made versions of swig, python and ROOT in directories labeled vc90, including at least those parts used by SCons to do builds. It's adequate for a mini-container with just the facilities package, enough to keep me going for quite a while.

January 8, 2009

Installed Express edition of VS 9.0. Since SCons as distributed does not specifically support it, plan is to

1. add support for VS 9.0
2. integrate work done by Tracy into the SCons msvs builder to improve project files part-way
3. make more comprehensive improvements to msvs builder to get all files desired into project without hard-coding and to make multi-project solution files.

Mid-December 2008

SCons as distributed supports builds with Visual Studio 2003 (and should also with Studio 2005 but I haven't tried it), but generates project and solution files which are inadequate in several respects. Since format of project files changed substantially after VS 2003 and since Windows users would like to move forward, the decision has been made to work on project files, etc., only for VS 2008 (aka version 9.0).